UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P O Box 1450
Alexandria, Virginia 22313-1450
www uspto gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/041,671	01/10/2002	Allon Adir	ADIR2	2977

1444 7590 05/25/2010
BROWDY AND NEIMARK, P.L.L.C.
624 NINTH STREET, NW
SUITE 300
WASHINGTON, DC 20001-5303

EXAMINER	
PATEL, SHAMBHAVI K	

ART UNIT	PAPER NUMBER
2128	

MAIL DATE	DELIVERY MODE
05/25/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)
	10/041,671	ADIR, ALLON
	Examiner	**Art Unit**
	SHAMBHAVI PATEL	2128

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 February 2010*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *27-38,40-51 and 71-78* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *27-38,40-51 and 71-78* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02 January 2002* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. Claims 27-38, 40-51, and 71-78 have been presented for examination.

Response to Arguments

2. In view of Applicant's amendments, the 35 U.S.C. 112 rejection is withdrawn.

3. Applicant's arguments regarding the prior art rejection have been fully considered but they are moot in view of the new grounds of rejection presented below.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

The factual inquiries set forth in Graham v. John Deere Co., 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

4. Claims 27-38 and 40-51 are rejected under 35 U.S.C. 103(a) as being unpatentable over Genie ("Genesys-MP: User's Guide) in view of **Saha ("A Simulation Based Approach to Architectural Verification of Multiprocessor Systems").**

Regarding claim 27:

Genie discloses a method for validating a processor design by simulating program execution, comprising the steps of

a. identifying a resource, comprising mutually dependent non-adjacent resources having non-contiguous addresses **(section 2.11.3: shared memory is divided into separate addresses/bytes which are mutually dependent since together they form the shared memory shown in figure 1, and discloses memory locations 1001 and 1003, which are non-contiguous addresses)** that may be accessed by a test program that includes a first simulated process and a second simulated process **(figure 1 "shared memory")**
b. identifying a set of value-lists, each value-list thereof containing permissible values of said resource **(section 2.11: 2nd paragraph: group of results for all possible orders in which instructions are interleaved)**
c. associating a set of non-unique values for said resource **(section 2.11: 2nd paragraph: group of results for all possible orders in which instructions are interleaved)**
d. executing said test program by executing a first sequence of instructions in said first simulated process and while performing said step of executing said first sequence, executing a second sequence of instructions in said second simulated process **(section 2.11.1: command for executing test cases)**
e. wherein said resource is accessed by at least one of said first simulated process and said second simulated process **(section 2.11: two processes can access the same memory at the same time)**, and wherein upon completion of said steps of executing said first sequence and executing said second sequence, a member of said set of non-unique values is required to be present in said resource **(section 2.11: group of possible results)**

Genie does not explicitly disclose creating, by the computer executing said test program, a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier and validating the processor design if a content of said resource is equal to a

member of one of said set of tagged value-lists. **Saha teaches** multi-processor validation **(abstract; introduction)** including creating a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier **(section 2: method uses valid data sets and the use of uniquely colored data for stores),** replacing said set of value-lists by said set of tagged value-lists **(section 2.2 3rd paragraph: updating rules)** and validating the processor design if a content of said resource is equal to a member of one of said set of tagged value-lists **(section 2.3: error detection).** At the time of the invention, it would have been obvious to one of ordinary skill in the art to combine the teachings of Genie and Saha because the method disclosed in Saha refers to true sharing, and testing with true sharing exposes more bugs by putting more stress on the coherency protocol implementations **(Saha: Introduction).**

Regarding claim 28:

Genie discloses the method according to claim 1, wherein said resource is a memory resource **(figure 1: shared memory)**

Regarding claim 29:

Genie discloses the method according to claim 1, wherein said resource is a register **(section 2.1.5 "memory sharing rules": register)**

Regarding claim 30:

Genie discloses the method according to claim 1, wherein said set of non-unique values is a set of value-lists **(section 2.11.5 "multiple results cards").**

Regarding claim 31:

Genie discloses the method according to claim 4, wherein said resource comprises a first adjacent resource and a second adjacent resource **(figure 1: shared memory contains multiple registers; section 2.1.5)**, having a contiguous address with the first adjacent resource **(section 2.11.3: 1001 and 1002)** and each member of said set of

list of values comprises a first value and a second value **(section 2.11.5 "multiple results cards")**, said first value being a permissible value of said first adjacent resource, and said second value being a permissible values of said second adjacent resource **(section 2.11.5 "multiple results cards": possible results)**, and verifying further comprising identifying a valid member of said set of list of values, by verifying an equality between a content of said first and second adjacent resource and said first and second value of said valid member **(section 2.11.5: testcase)**

Regarding claim 32:

Genie discloses the method according to claim 1, wherein said resource comprises a first resource and a second resource **(figure 1: shared memory contains multiple registers; section 2.1.5)** and said set of non-unique values comprises a first set of non-unique values that is associated with said first resource, and a second set of non-unique values that is associated with said second resource **(section 2.11.5 "multiple results cards")**, further comprising associating a first member of said first set of non-unique values with a second member of said second set of non-unique values **(sections 2.11.3 and 2.11.4: collisions)**; and verifying an equality between said first resource and said first member; and verifying an equality between said second resource and said second member **(section 2.11.5: testcase).**

Regarding claim 33:

Genie discloses the method according to claim 6, wherein said step of associating said first member is performed by tagging said first member and said second member with a common combination identifier of a particular outcome of said test program **(section 2.11.3: results contained in same list).**

Regarding claim 34:

Genie discloses the method according to claim 6, wherein said first member of said first set of non-unique values comprises a first list of values, and said second member of said second set of non-unique values comprises a second list of values **(section 2.11.5 "multiple results cards")**, respective elements of said first list of values being permissible values of said first resource and adjacent resources having contiguous addresses **(section 2.11.3: 1001**

and 1002) thereof, and respective elements of said second list of values being permissible values of said second resource and adjacent resources thereof **(section 2.11.5 "multiple results cards": possible results)**, verifying an equality between a content of said first resource and adjacent resources thereof with corresponding elements of said first list of values; and verifying an equality between a content of said second resource and adjacent resources thereof with corresponding elements of said second list of values **(section 2.11.5: testcase).**

Regarding claim 35:

Genie discloses the method according to claim 1, wherein said step of associating said set of non-unique values is performed prior to said step of executing said first sequence of instructions **(section 2.11.5 "multiple results cards").**

Regarding claim 36:

Genie discloses the method according to claim 9, further comprising the steps of: defining a results section in said test program, and entering all permissible values assumable by said resource and an identifier of said resource in an entry of said results section **(section 2.11.5 "multiple results cards": all possible result values).**

Regarding claim 37:

Genie discloses the method according to claim 1, wherein said step of executing said test program further comprises the steps of: generating said first sequence and said second sequence to define generated instructions **(section 3.5),** simulating one of said generated instructions in said first simulated process and said second simulated process **(section 2.11.1: command for executing test cases)** maintaining a store that contains a set of values that are assumable in said resource during said step of simulating said one of said generated instructions **(section 2.11.5 "multiple results cards")** and thereafter determining whether said store contains non-unique values **(section 2.11.5: testcase).**

Regarding claim 38:

Genie discloses the method according to claim 11, further storing in said store first values of said resource during accesses thereof by said first simulated process and storing in said store second values of said resource by said second simulated process during accesses thereof **(section 2.12: processor 1 and processor 2)**, wherein said first values comprise first written values, and said second values comprise second written values **(section 2.12: instructions store load)** and; identifying in said store a last value written to said resource in said step of simulating said one of said generated instructions **(section 2.12: instructions).**

Regarding claim 40:

Genie discloses the method according to claim 1, further comprising the step of establishing a synchronization barrier for said first simulated process and said second simulated process **(section 2.9 synchronization protocol).**

Regarding claim 41:

Genie discloses the method according to claim 1, further comprising the steps of biasing generation of said test program to promote collisions of memory accessing instructions that are executed by said first simulated process and said second simulated process **(section 2.13: memory biasing).**

Regarding claim 42:

Genie discloses a method of verification of an architecture by simulation, comprising the steps of:

a. defining a program input to a test generator **(section 2.1 file)**
b. generating a test program responsive to said program input **(section 3.1)** said test program including a list of resource initializations **(section 3.1 processor)**, a list of instructions **(sections 2.11.3 and 2.11.4: collisions)**, and a list of predicted resource results comprising mutually dependent non-adjacent resources having non-contiguous addresses **(section 2.11.3: shared memory is divided into separate addresses/bytes which are mutually dependent since together they form the shared memory shown in figure 1, and discloses memory locations 1001 and 1003, which are non-contiguous addresses)**, wherein at least one member of said list

of predicted resource results comprises value-list containing permissible values of a resource **(section 2.11: group of possible results)**

c. simulating an execution of said test program using a plurality of simultaneously executing processes **(section 2.11: two processes can access the same memory at the same time)**

Genie does not explicitly disclose creating, by the computer executing said test program, a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier and validating the processor design if a content of said resource is equal to a member of one of said set of tagged value-lists. **Saha teaches** multi-processor validation **(abstract; introduction)** including creating a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier **(section 2: method uses valid data sets and the use of uniquely colored data for stores),** replacing said set of value-lists by said set of tagged value-lists **(section 2.2 3rd paragraph: updating rules)** and validating the processor design if a content of said resource is equal to a member of one of said set of tagged value-lists **(section 2.3: error detection).** At the time of the invention, it would have been obvious to one of ordinary skill in the art to combine the teachings of Genie and Saha because the method disclosed in Saha refers to true sharing, and testing with true sharing exposes more bugs by putting more stress on the coherency protocol implementations **(Saha: Introduction).**

Regarding claim 43:

Genie discloses the method according to claim 16, wherein said list of predicted resource results comprises predicted results of adjacent resources having contiguous addresses **(section 2.11.3: 1001 and 1002)**, wherein said adjacent resources are mutually dependent, and said step of verifying said actual resource result is performed by verifying each of said adjacent resources **(figure 1: shared memory contains multiple registers; section 2.1.5)**.

Regarding claim 44:

Genie discloses the method according to claim 17, wherein said adjacent resources are memory resources **(figure 1: shared memory)**.

Regarding claim 45:

Genie discloses the method according to claim 17, wherein said adjacent resources are registers **(section 2.1.5 "memory sharing rules": register).**

Regarding claim 46:

Genie discloses the method according to claim 16, further comprising the step of establishing a synchronization barrier for said simultaneously executing processes **(section 2.9 synchronization protocol).**

Regarding claim 47:

Genie discloses the method according to claim 1, further comprising the steps of biasing generation of said test program to promote collisions of memory accessing instructions that are executed by said first simulated process and said second simulated process **(section 2.13: memory biasing).**

Regarding claim 48:

Genie discloses the method according to claim 42, further identifying a combination of said mutually dependent non-adjacent resources by tagging corresponding members of said list of predicted resource results with a unique combination identifier of a particular outcome of said test program so as to define commonly tagged lists of values of predicted resource results **(section 2.11.3: results contained in same list)**; and said step of verifying said actual resource result is performed by verifying that resources of said combination have actual results that are equal to a member of a corresponding one of said commonly tagged lists of values **(section 2.11.5: testcase).**

Regarding claim 49:

Genie discloses a method of predicting non-unique results by simulating a system design, comprising the steps of:

a. defining a program input to a test generator **(section 2.1 file)**

b. generating a test program responsive to said program input**(section 3.1)** said test program including a list of resource initializations **(section 3.1 processor)**, a list of instructions **(sections 2.11.3 and 2.11.4: collisions)**, and a list of predicted resource results comprising mutually dependent non-adjacent resources having non-contiguous addresses **(section 2.11.3: shared memory is divided into separate addresses/bytes which are mutually dependent since together they form the shared memory shown in figure 1, and discloses memory locations 1001 and 1003, which are non-contiguous addresses)**, wherein at least one member of said list of predicted resource results comprises value-list containing permissible values of a resource **(section 2.11: group of possible results)**

c. simulating an execution of a single instruction of said test program by a first process of said test program **(section 2.11: instruction simulation)**

d. calculating possible values of target resources of said single instruction **(section 2.11.5: testcase)**

Genie does not explicitly disclose creating, by the computer executing said test program, a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier and validating the processor design if a content of said resource is equal to a member of one of said set of tagged value-lists. **Saha teaches** multi-processor validation **(abstract; introduction)** including creating a set of tagged value-lists by tagging members of a list of predicted results with a combination identifier identifying a particular outcome of said test program, each of said set of tagged value-lists comprising a value list from said set of value-lists and said combination identifier **(section 2: method uses valid data sets and the use of uniquely colored data for stores),** replacing said set of value-lists by said set of tagged value-lists **(section 2.2 3rd paragraph: updating rules)** and validating the processor design if a content of said resource is equal to a member of one of said set of tagged value-lists **(section 2.3: error detection).** At the time of the invention, it would have been obvious to one of ordinary skill in the art to combine the teachings of Genie and Saha because the method disclosed in Saha refers to true sharing, and testing with true sharing exposes more bugs by putting more stress on the coherency protocol implementations **(Saha: Introduction)..**

Regarding claim 50:

Genie discloses the method according to claim 21, further comprising the steps of: performing said step of simulating an execution by a second process of said test program **(section 2.11.1: command for executing test cases),** maintaining lists of written values that are written to said target resources of said single instruction by said first process and said second process **(section 3.5: write)**; and determining respective last values in said lists of written values **(section 2.11.5: testcase).**

Regarding claim 51:

Genie discloses the computer software product according to claim 50, the method further comprising the steps of:

a. prior to performing said steps of simulating said execution by said first process and of simulating said execution by said second process memorizing an initial simulated state of said test program **(section 3.1 configuration)**
b. maintaining lists of read values that are read from source resources of said single instruction **(section 3.5: write)**
c. identifying a member of said lists of read values so as to define an identified member **(section 2.11.5: testcase)**
d. restoring said initial simulated state of said test program **(section 3.1 configuration)**
e. performing said step of simulating said execution a second time, and reading said identified member, using an associated process other than the first and second processes thereof **(section 2.11.1: command for executing test cases)**

Regarding claim 72:

Genie discloses the computer software product according to claim 42, wherein said non-adjacent resources are memory resources **(figure 1: shared memory).**

5. Claims 71, 73-78 are rejected under 35 U.S.C. 103(a) as being unpatentable over Genie ("Genesys-MP: User's Guide) in view of **Saha ("A Simulation Based Approach to Architectural Verification of Multiprocessor Systems")** in view of **Luo ("Development and Validation of a Hierarchical Memory Model Incorporating CPU- and Memory- Operation Overlap").**

Regarding claims 71, 73-78:

Genie discloses memories where addresses are indexes **(section 2.11.3: 1001, 1002...). Genie does not explicitly disclose** resources comprising different types, such as registers. **Luo discloses** memory verification **(abstract; Introduction)** of non-adjacent, non-contiguous resources **(figure 1: distributed processors and memories)** wherein the resources comprise different types such as memories and registers **(figure 1; section 2.1: 1st and 2nd paragraphs: local memories and memories of processors include registers).** At the time of the invention, it would have been obvious to one of ordinary skill in the art to combine the teachings of Genie, Saha and Luo in order to implement studies that promise insight for engineers in future architectural design improvements and will provide information that software engineers can use to improve code performance in scalable environments **(Luo: Introduction).**

Conclusion

6. **Examiner's Remarks:** Examiner has cited particular columns and line numbers in the references applied to the claims above for the convenience of the applicant. Although the specified citations are representative of the teachings of the art and are applied to specific limitations within the individual claim, other passages and figures may apply as well. It is respectfully requested from the applicant in preparing responses, to fully consider the references in their entirety as potentially teaching all or part of the claimed invention, as well as the context of the passage as taught by the prior art or disclosed by the Examiner. In the case of amending the claimed invention, Applicant is respectfully requested to indicate the portion(s) of the specification which dictate(s) the structure relied on for proper interpretation and also to verify and ascertain the metes and bounds of the claimed invention.

7. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Shambhavi Patel whose telephone number is (571) 272-5877. The examiner can normally be reached on Monday-Friday, 8:00 am – 4:30 pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kamini Shah can be reached on (571) 272-2279. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

SKP

/Kamini S Shah/
Supervisory Patent Examiner, Art Unit 2128